CPE301 – SPRING 2022

Design Assignment 1

Student Name: Ernesto Ibarra-Ayala
Student #: 2001211571
Student Email: ibarre3@unlv.nevada.edu
Primary Github address: https://github.com/ErnestoIbarra333
Directory: https://github.com/ErnestoIbarra333/ErnestoIbarra.git

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

So far, we only used Atmel Studios and nothing else just yet. We will be using the atmega328p board soon.

2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A

No initial code given

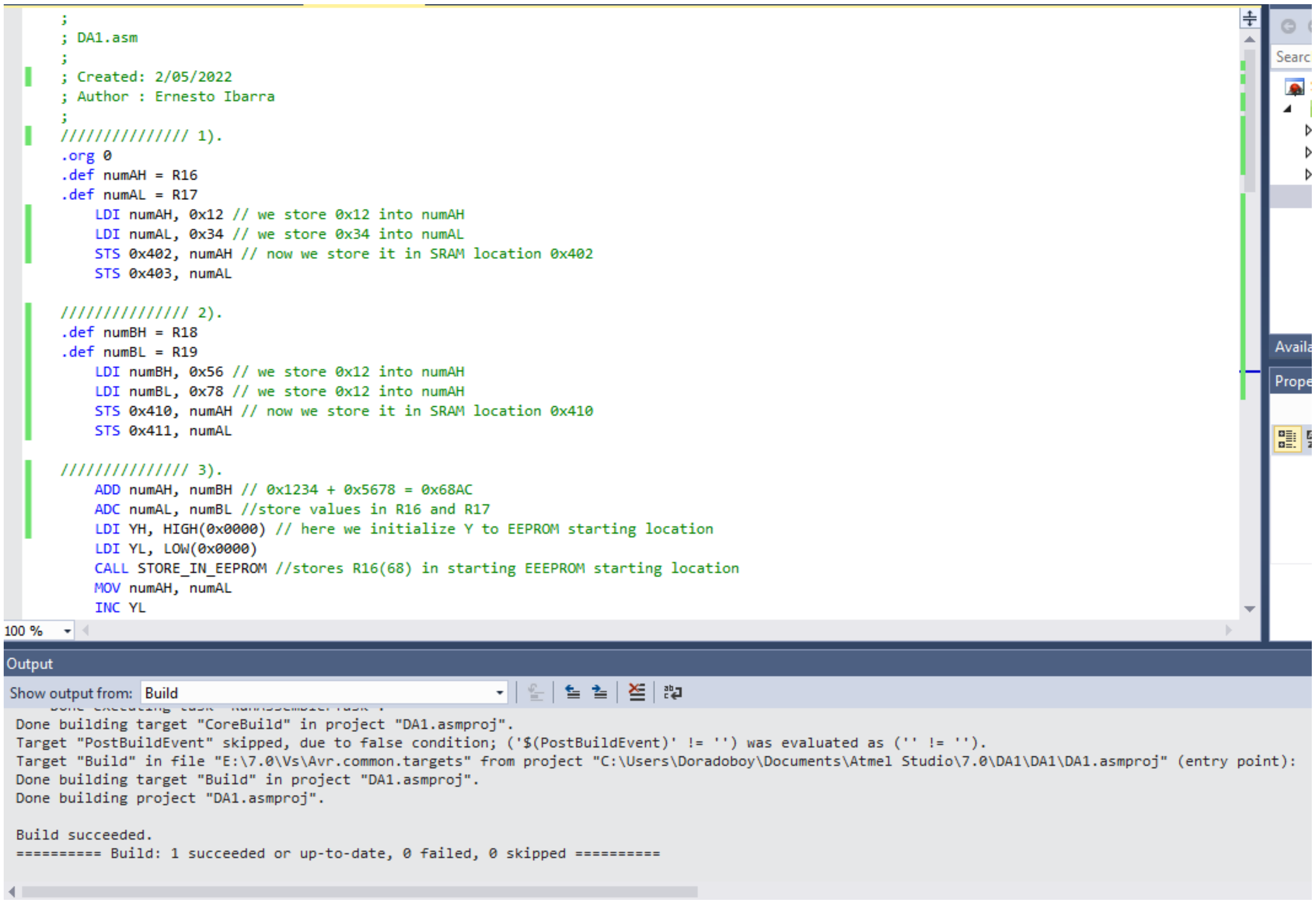
3. DEVELOPED MODIFIED CODE OF TASK 1/2/3/4

Here is my code screenshot as well as the actual code copied and pasted. I also put a screenshot of my code building successfully.

```
;
; DA1.asm
;
; Created: 2/05/2022
; Author : Ernesto Ibarra
;
/////////////// 1).
.org 0
.def numAH = R16
.def numAL = R17
       LDI numAH, 0x12 // we store 0x12 into numAH
       LDI numAL, 0x34 // we store 0x34 into numAL
       STS 0x402, numAH // now we store it in SRAM location 0x402
       STS 0x403, numAL

/////////////// 2).
.def numBH = R18
.def numBL = R19
       LDI numBH, 0x56 // we store 0x12 into numAH
       LDI numBL, 0x78 // we store 0x12 into numAH
       STS 0x410, numAH // now we store it in SRAM location 0x410
       STS 0x411, numAL
```

```
/////////////// 3).
       ADD numAH, numBH // 0x1234 + 0x5678 = 0x68AC
       ADC numAL, numBL //store values in R16 and R17
       LDI YH, HIGH(0x0000) // here we initialize Y to EEPROM starting location
       LDI YL, LOW(0x0000)
       CALL STORE_IN_EEPROM //stores R16(68) in starting EEEPROM starting location
       MOV numAH, numAL
       INC YL
       CALL STORE_IN_EEPROM // stores R17(AC) in the next EEPROM location
       NOP

/////////////// 4).
.EQU STARTADDS = 0x500

.def tmp = R20 // tmp variable to hold values
.def count = R22 // count for the loop
.def sumcount = R23 // count to add the numbers
.def sumH = R24 // here we will store the values
.def sumL = R25
       LDI R21, HIGH(RAMEND) // here we initialize the stack
       OUT SPH, R21 // we are using R21
       LDI R21, LOW(RAMEND)
       OUT SPL, R21

       LDI ZL, LOW(2*MYDATA1) // Here we let Z point to our Data
       LDI ZH, HIGH(2*MYDATA1)
       LDI XL, LOW(STARTADDS) // X will point to our address to store it in
       LDI XH, HIGH(STARTADDS)
       ldi count, 20
       ldi sumcount, 10

LOOP1: // Here we will store 10 in program memory then retrieve them and store them in
SRAM using X pointer
       lpm tmp, Z+ // here we load Z into tmp
       PUSH tmp //PUSH it so we can later add all the numbers easily
       ST  X+, tmp // loads into SRAM location 0x500
       DEC count
       brne LOOP1
LOOP2: // Here we will add all the 10 16 bit numbers and store them in SRAM starting
location 0x406
       POP numAH //High byte
       POP numAL //Low byte
       ADD sumH, numAH // Here sumH and sumL are keeping stored all the addition
       ADC sumL, numAL
       DEC sumcount
       brne LOOP2
       STS 0x406, sumH // after we have added 0x0910+0x0911+0x0912 .... and we get a
final value of 5ACD
       STS 0x407, sumL

END: JMP END // program ends

.ORG 0x1000
MYDATA1: .dw 0x0910,0x0911,0x0912,0x0913,0x0914,0x0915,0x0916,0x0917,0x0918,0x0919
```

```
STORE_IN_EEPROM: // Store function for EEPROM
        SBIC EECR, EEPE
        RJMP STORE_IN_EEPROM
        OUT EEARH, YH
        OUT EEARL, YL
        OUT EEDR, numAH
        SBI EECR, EEMPE
        SBI EECR, EEPE
        RET
```



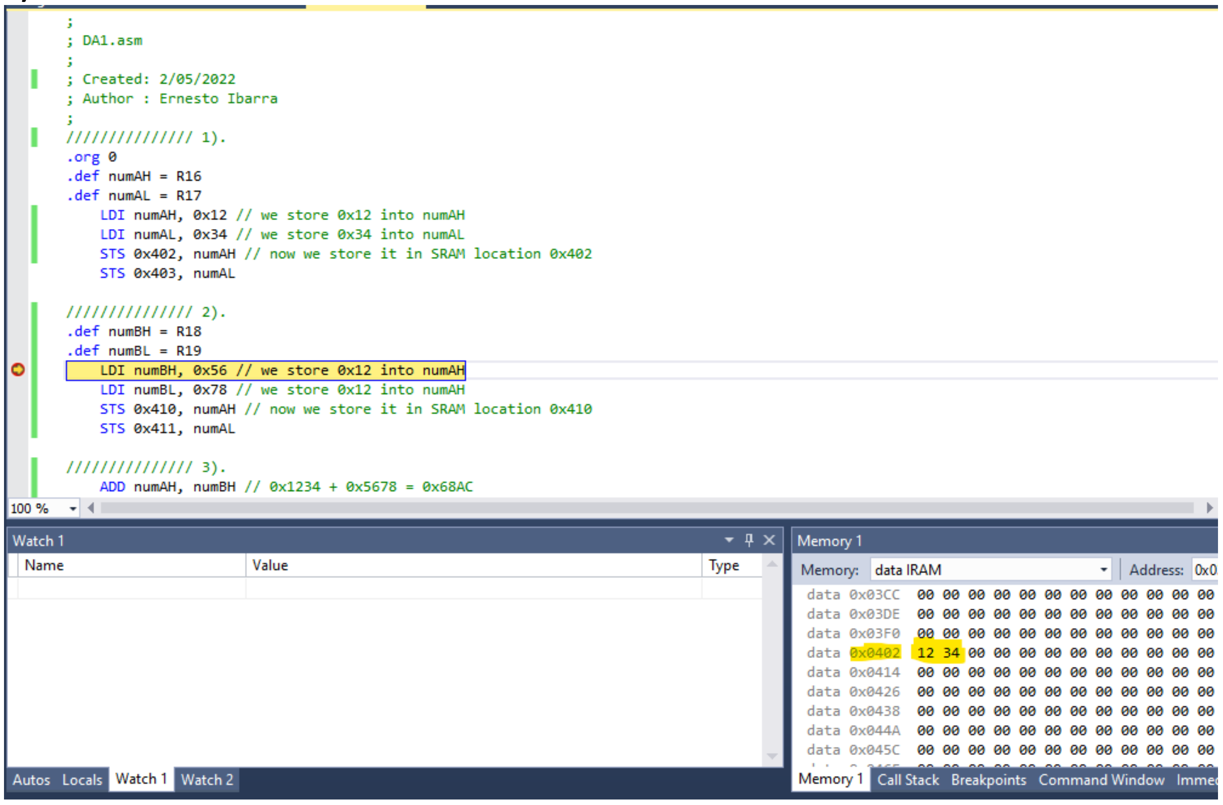
```
;
; DA1.asm
;
; Created: 2/05/2022
; Author : Ernesto Ibarra
;
/////////////// 1).
.org 0
.def numAH = R16
.def numAL = R17
    LDI numAH, 0x12 // we store 0x12 into numAH
    LDI numAL, 0x34 // we store 0x34 into numAL
    STS 0x402, numAH // now we store it in SRAM location 0x402
    STS 0x403, numAL

/////////////// 2).
.def numBH = R18
.def numBL = R19
    LDI numBH, 0x56 // we store 0x12 into numAH
    LDI numBL, 0x78 // we store 0x12 into numAH
    STS 0x410, numAH // now we store it in SRAM location 0x410
    STS 0x411, numAL

/////////////// 3).
    ADD numAH, numBH // 0x1234 + 0x5678 = 0x68AC
    ADC numAL, numBL //store values in R16 and R17
    LDI YH, HIGH(0x0000) // here we initialize Y to EEPROM starting location
    LDI YL, LOW(0x0000)
    CALL STORE_IN_EEPROM //stores R16(68) in starting EEEPROM starting location
    MOV numAH, numAL
    INC YL
```

```
Done building target "CoreBuild" in project "DA1.asmproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "E:\7.0\Vs\Avr.common.targets" from project "C:\Users\Doradoboy\Documents\Atmel Studio\7.0\DA1\DA1\DA1.asmproj" (entry point):
Done building target "Build" in project "DA1.asmproj".
Done building project "DA1.asmproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

4. SCHEMATICS

5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

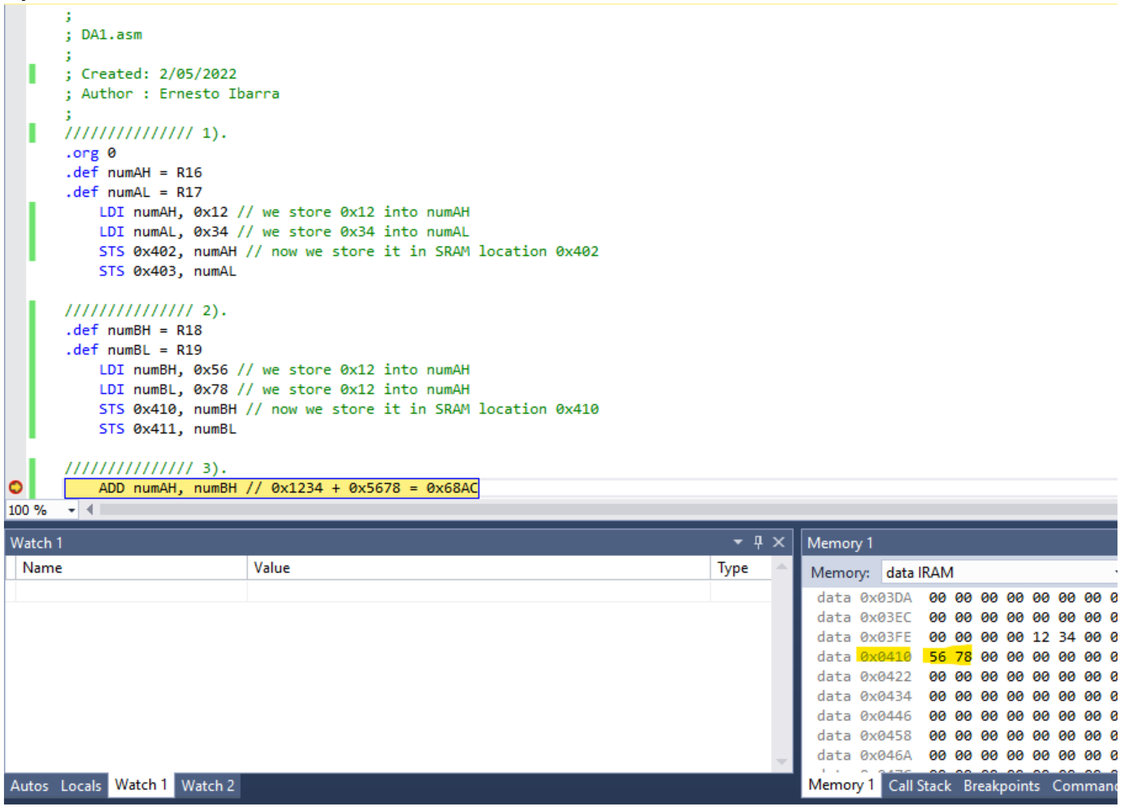
1).



```
;
; DA1.asm
;
; Created: 2/05/2022
; Author : Ernesto Ibarra
;
/////////////// 1).
.org 0
.def numAH = R16
.def numAL = R17
    LDI numAH, 0x12 // we store 0x12 into numAH
    LDI numAL, 0x34 // we store 0x34 into numAL
    STS 0x402, numAH // now we store it in SRAM location 0x402
    STS 0x403, numAL

/////////////// 2).
.def numBH = R18
.def numBL = R19
    LDI numBH, 0x56 // we store 0x12 into numAH
    LDI numBL, 0x78 // we store 0x12 into numAH
    STS 0x410, numAH // now we store it in SRAM location 0x410
    STS 0x411, numAL

/////////////// 3).
    ADD numAH, numBH // 0x1234 + 0x5678 = 0x68AC
```

```
data 0x03CC  00 00 00 00 00 00 00 00 00 00 00 00
data 0x03DE  00 00 00 00 00 00 00 00 00 00 00 00
data 0x03F0  00 00 00 00 00 00 00 00 00 00 00 00
data 0x0402  12 34 00 00 00 00 00 00 00 00 00 00
data 0x0414  00 00 00 00 00 00 00 00 00 00 00 00
data 0x0426  00 00 00 00 00 00 00 00 00 00 00 00
data 0x0438  00 00 00 00 00 00 00 00 00 00 00 00
data 0x044A  00 00 00 00 00 00 00 00 00 00 00 00
data 0x045C  00 00 00 00 00 00 00 00 00 00 00 00
```

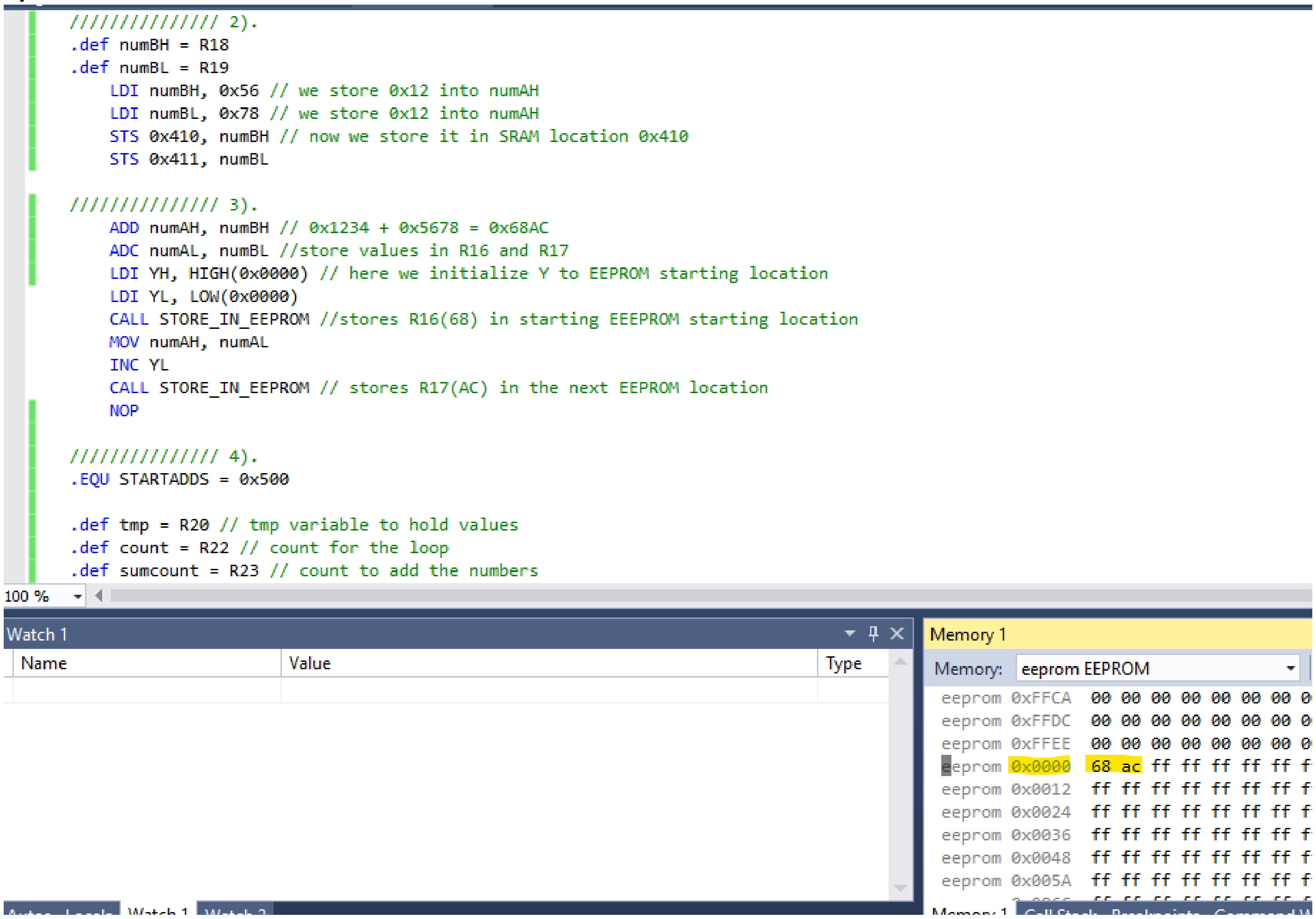
2).



```
;
; DA1.asm
;
; Created: 2/05/2022
; Author : Ernesto Ibarra
;
/////////////// 1).
.org 0
.def numAH = R16
.def numAL = R17
    LDI numAH, 0x12 // we store 0x12 into numAH
    LDI numAL, 0x34 // we store 0x34 into numAL
    STS 0x402, numAH // now we store it in SRAM location 0x402
    STS 0x403, numAL

/////////////// 2).
.def numBH = R18
.def numBL = R19
    LDI numBH, 0x56 // we store 0x12 into numAH
    LDI numBL, 0x78 // we store 0x12 into numAH
    STS 0x410, numBH // now we store it in SRAM location 0x410
    STS 0x411, numBL

/////////////// 3).
    ADD numAH, numBH // 0x1234 + 0x5678 = 0x68AC
```

```
data 0x03DA  00 00 00 00 00 00 00 0
data 0x03EC  00 00 00 00 00 00 00 0
data 0x03FE  00 00 00 00 12 34 00 0
data 0x0410  56 78 00 00 00 00 00 0
data 0x0422  00 00 00 00 00 00 00 0
data 0x0434  00 00 00 00 00 00 00 0
data 0x0446  00 00 00 00 00 00 00 0
data 0x0458  00 00 00 00 00 00 00 0
data 0x046A  00 00 00 00 00 00 00 0
```

3).



```
/////////////// 2).
.def numBH = R18
.def numBL = R19
    LDI numBH, 0x56 // we store 0x12 into numAH
    LDI numBL, 0x78 // we store 0x12 into numAH
    STS 0x410, numBH // now we store it in SRAM location 0x410
    STS 0x411, numBL

////////////// 3).
    ADD numAH, numBH // 0x1234 + 0x5678 = 0x68AC
    ADC numAL, numBL //store values in R16 and R17
    LDI YH, HIGH(0x0000) // here we initialize Y to EEPROM starting location
    LDI YL, LOW(0x0000)
    CALL STORE_IN_EEPROM //stores R16(68) in starting EEEPROM starting location
    MOV numAH, numAL
    INC YL
    CALL STORE_IN_EEPROM // stores R17(AC) in the next EEPROM location
    NOP

////////////// 4).
.EQU STARTADDS = 0x500

.def tmp = R20 // tmp variable to hold values
.def count = R22 // count for the loop
.def sumcount = R23 // count to add the numbers
```

100 %

Watch 1

Name	Value	Type

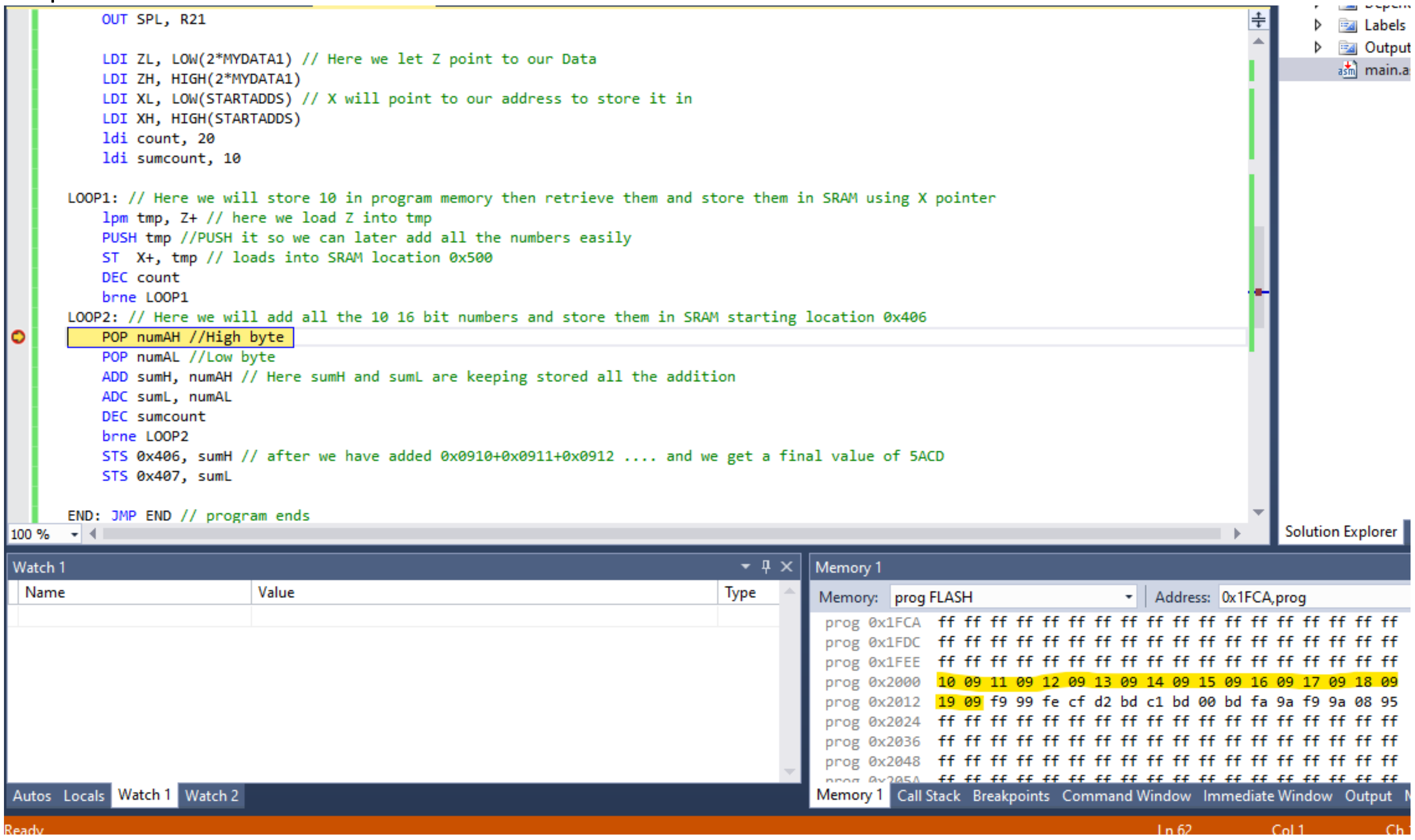
Memory 1

Memory: eeprom EEPROM

```
eeprom 0xFFCA  00 00 00 00 00 00 00 0
eeprom 0xFFDC  00 00 00 00 00 00 00 0
eeprom 0xFFEE  00 00 00 00 00 00 00 0
eeprom 0x0000  68 ac ff ff ff ff ff f
eeprom 0x0012  ff ff ff ff ff ff ff f
eeprom 0x0024  ff ff ff ff ff ff ff f
eeprom 0x0036  ff ff ff ff ff ff ff f
eeprom 0x0048  ff ff ff ff ff ff ff f
eeprom 0x005A  ff ff ff ff ff ff ff f
```

Autos Locals Watch 1 Watch 2
Memory 1 Call Stack Breakpoints Command W

4). First, we have part one of question 4 which is storing ten 16-bit numbers starting from 0x0910 into program memory. The location wasn't specified so I just chose 0x1000 to keep it simple.

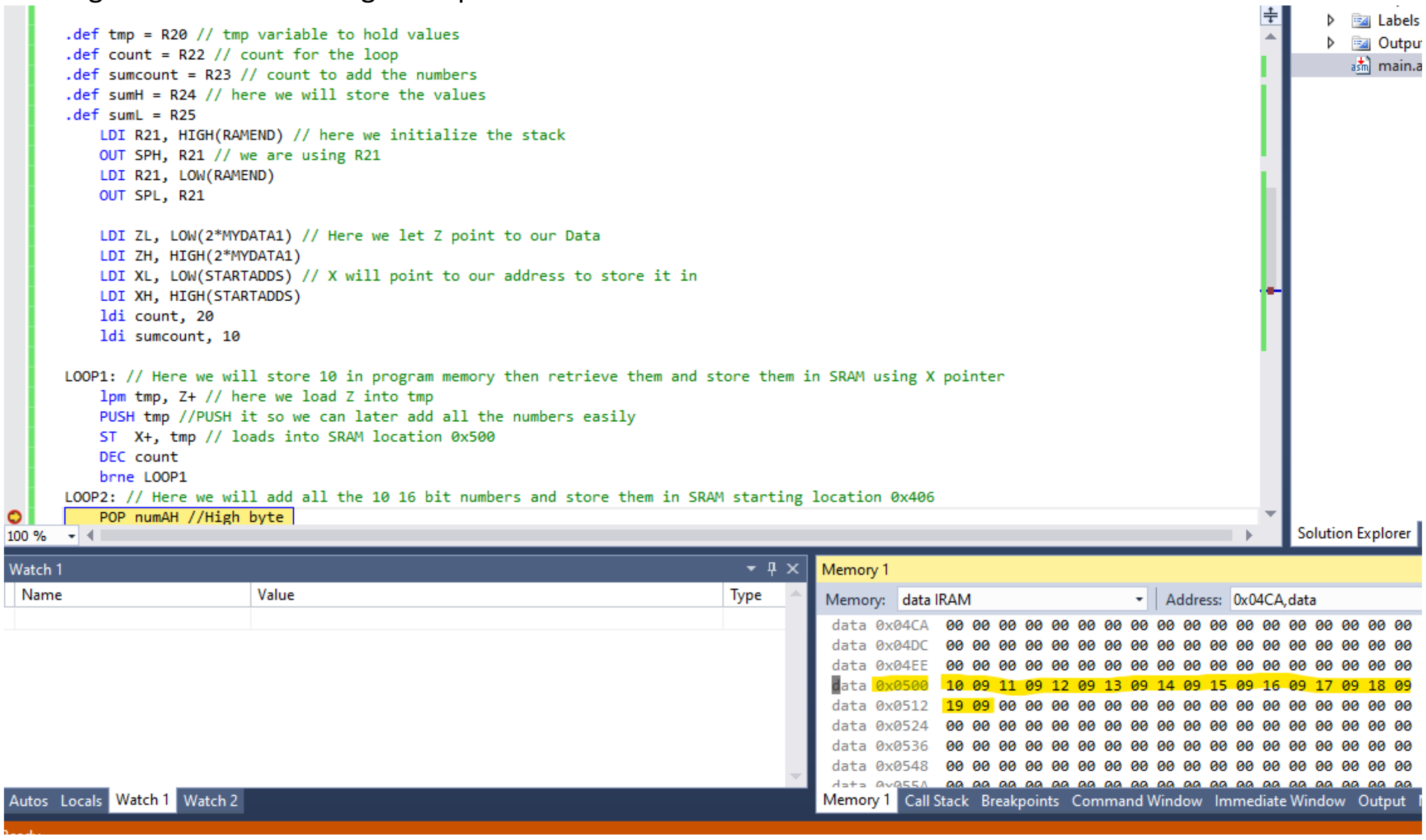


```
    OUT SPL, R21

    LDI ZL, LOW(2*MYDATA1) // Here we let Z point to our Data
    LDI ZH, HIGH(2*MYDATA1)
    LDI XL, LOW(STARTADDS) // X will point to our address to store it in
    LDI XH, HIGH(STARTADDS)
    ldi count, 20
    ldi sumcount, 10

LOOP1: // Here we will store 10 in program memory then retrieve them and store them in SRAM using X pointer
    lpm tmp, Z+ // here we load Z into tmp
    PUSH tmp //PUSH it so we can later add all the numbers easily
    ST  X+, tmp // loads into SRAM location 0x500
    DEC count
    brne LOOP1
LOOP2: // Here we will add all the 10 16 bit numbers and store them in SRAM starting location 0x406
    POP numAH //High byte
    POP numAL //Low byte
    ADD sumH, numAH // Here sumH and sumL are keeping stored all the addition
    ADC sumL, numAL
    DEC sumcount
    brne LOOP2
    STS 0x406, sumH // after we have added 0x0910+0x0911+0x0912 .... and we get a final value of 5ACD
    STS 0x407, sumL

END: JMP END // program ends
```

```
Memory: prog FLASH    Address: 0x1FCA,prog
prog 0x1FCA  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x1FDC  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x1FEE  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x2000  10 09 11 09 12 09 13 09 14 09 15 09 16 09 17 09 18 09
prog 0x2012  19 09 f9 99 fe cf d2 bd c1 bd 00 bd fa 9a f9 9a 08 95
prog 0x2024  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x2036  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x2048  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
```

Labels
Output
main.a
100 %
Solution Explorer
Watch 1
Name
Value
Type
Memory 1
Autos Locals Watch 1 Watch 2
Memory 1 Call Stack Breakpoints Command Window Immediate Window Output
Ready
Ln 62
Col 1

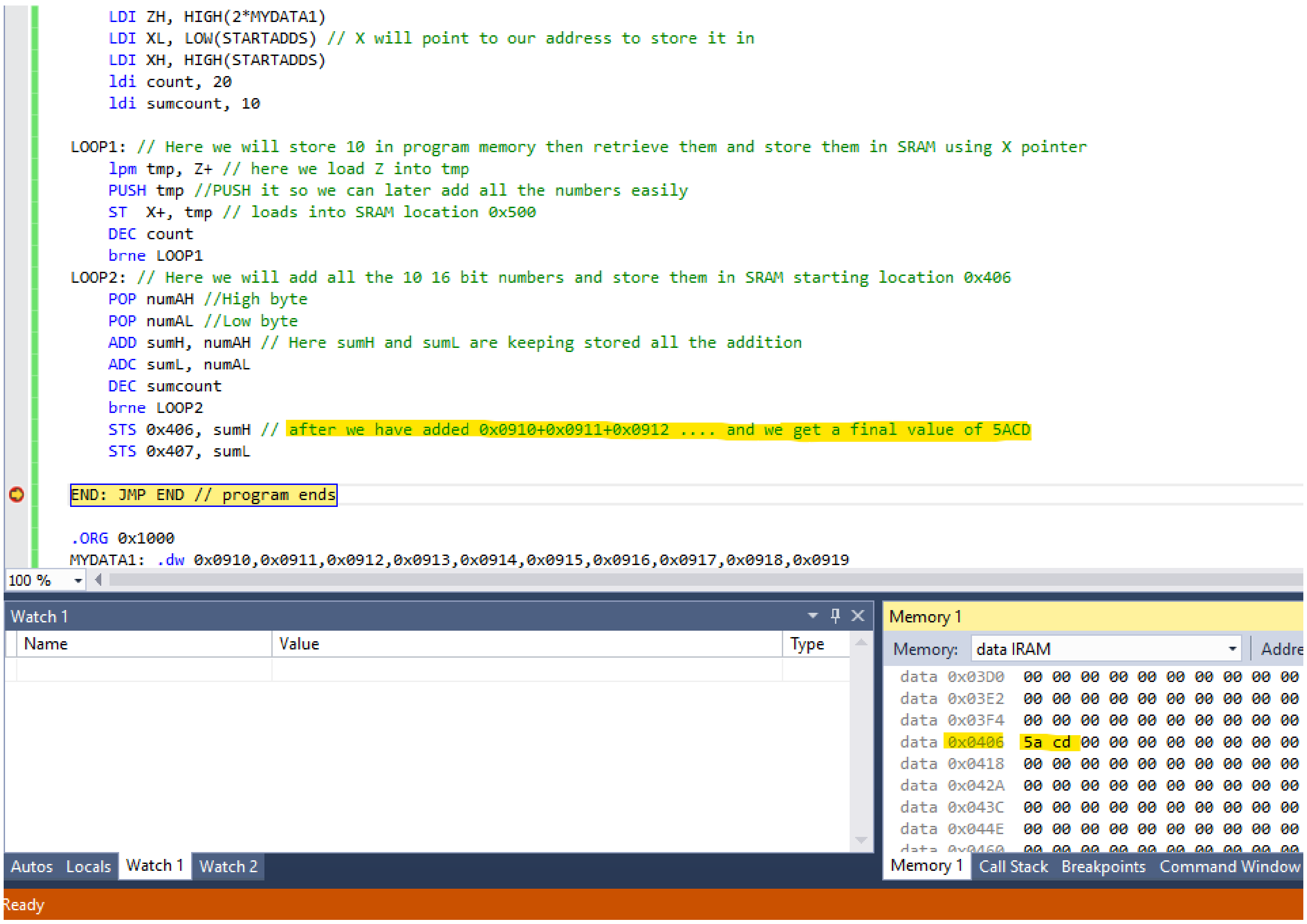
4). Here is part two, now we have to retrieve those 10 numbers and store them in SRAM starting location 0x500 using the X pointer.



```
.def tmp = R20 // tmp variable to hold values
.def count = R22 // count for the loop
.def sumcount = R23 // count to add the numbers
.def sumH = R24 // here we will store the values
.def sumL = R25
    LDI R21, HIGH(RAMEND) // here we initialize the stack
    OUT SPH, R21 // we are using R21
    LDI R21, LOW(RAMEND)
    OUT SPL, R21

    LDI ZL, LOW(2*MYDATA1) // Here we let Z point to our Data
    LDI ZH, HIGH(2*MYDATA1)
    LDI XL, LOW(STARTADDS) // X will point to our address to store it in
    LDI XH, HIGH(STARTADDS)
    ldi count, 20
    ldi sumcount, 10

LOOP1: // Here we will store 10 in program memory then retrieve them and store them in SRAM using X pointer
    lpm tmp, Z+ // here we load Z into tmp
    PUSH tmp //PUSH it so we can later add all the numbers easily
    ST  X+, tmp // loads into SRAM location 0x500
    DEC count
    brne LOOP1
LOOP2: // Here we will add all the 10 16 bit numbers and store them in SRAM starting location 0x406
    POP numAH //High byte
```

```
data 0x04CA  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x04DC  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x04EE  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0500  10 09 11 09 12 09 13 09 14 09 15 09 16 09 17 09 18 09
data 0x0512  19 09 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0524  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0536  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0548  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

4). Here is part three, now we have to sum up those 10 numbers and store them into SRAM starting location 0x406. I used a stack to make it easier to pop the numbers and add them.



```
    LDI ZH, HIGH(2*MYDATA1)
    LDI XL, LOW(STARTADDS) // X will point to our address to store it in
    LDI XH, HIGH(STARTADDS)
    ldi count, 20
    ldi sumcount, 10

LOOP1: // Here we will store 10 in program memory then retrieve them and store them in SRAM using X pointer
    lpm tmp, Z+ // here we load Z into tmp
    PUSH tmp //PUSH it so we can later add all the numbers easily
    ST  X+, tmp // loads into SRAM location 0x500
    DEC count
    brne LOOP1
LOOP2: // Here we will add all the 10 16 bit numbers and store them in SRAM starting location 0x406
    POP numAH //High byte
    POP numAL //Low byte
    ADD sumH, numAH // Here sumH and sumL are keeping stored all the addition
    ADC sumL, numAL
    DEC sumcount
    brne LOOP2
    STS 0x406, sumH // after we have added 0x0910+0x0911+0x0912 .... and we get a final value of 5ACD
    STS 0x407, sumL

END: JMP END // program ends

.ORG 0x1000
MYDATA1: .dw 0x0910,0x0911,0x0912,0x0913,0x0914,0x0915,0x0916,0x0917,0x0918,0x0919
```

```
Memory: data IRAM
data 0x03D0  00 00 00 00 00 00 00 00 00 00
data 0x03E2  00 00 00 00 00 00 00 00 00 00
data 0x03F4  00 00 00 00 00 00 00 00 00 00
data 0x0406  5a cd 00 00 00 00 00 00 00 00
data 0x0418  00 00 00 00 00 00 00 00 00 00
data 0x042A  00 00 00 00 00 00 00 00 00 00
data 0x043C  00 00 00 00 00 00 00 00 00 00
data 0x044E  00 00 00 00 00 00 00 00 00 00
```

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

7. VIDEO LINKS OF EACH DEMO

8. GITHUB LINK OF THIS DA

https://github.com/ErnestoIbarra333/ErnestoIbarra.git

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
Ernesto Ibarra-Ayala